

Module: Systèmes Embarqués

Chapitre 5:
Les Circuits Logiques Programmables & les ASICs

El Kefi HLEL

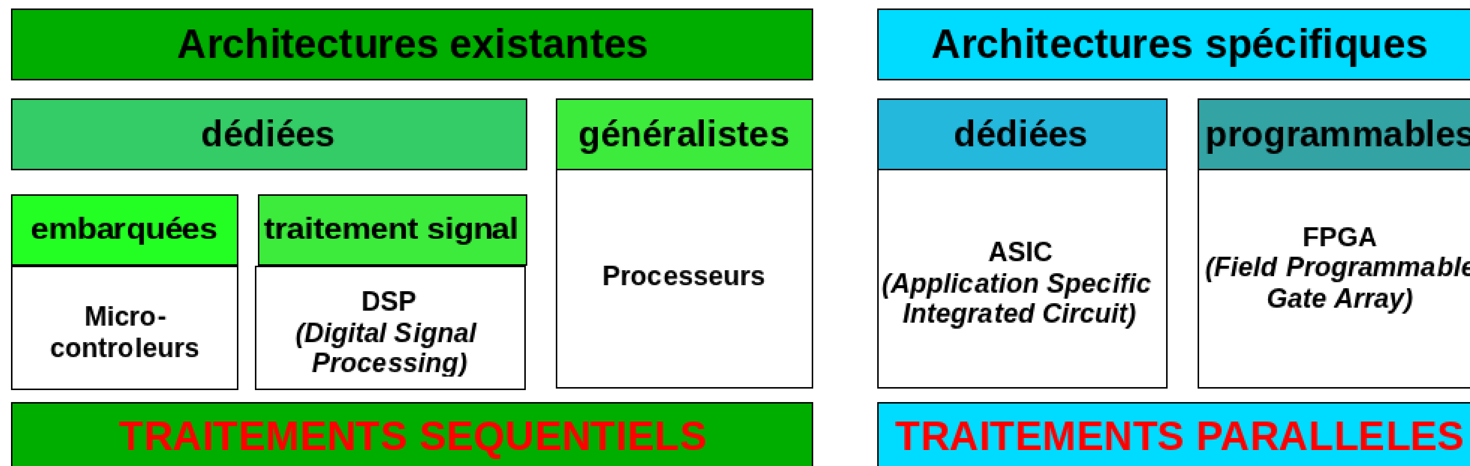
Février 2014

INTRODUCTION

Système Embarqué

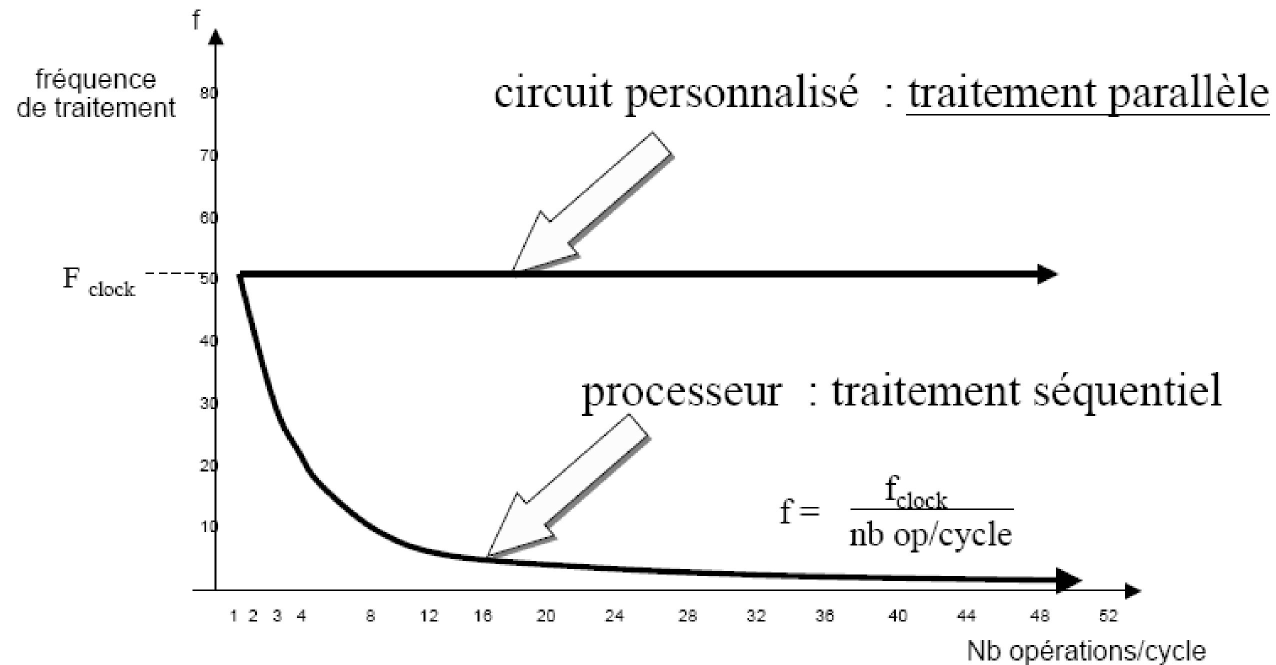
Pour réaliser un système embarqué:

- Processeur, Microcontrôleur, DSP
  - Avantages: temps de développement réduit, souplesse de programmation
  - Désavantages: temps d'exécution, coût structure minimum CPU-RAM-ROM-E/S
- ASIC
  - Avantages: exécution très rapide, consommation optimisée
  - Désavantages: manque de flexibilité, temps et coûts de développement
- Logique programmable
  - Avantages: flexibilité, exécution rapide, parallélisme
  - Désavantages: plus lent que les ASIC, développement plus long que processeur



- Un **circuit logique programmable**, ou **réseau logique programmable**, est un circuit intégré logique qui peut être « reprogrammé » après sa fabrication
- Il serait impropre de parler de programmation au sens logiciel (contrairement à un microprocesseur, il n'exécute aucune ligne de code). Dans ce cas, il est mieux de parler de « **reconfiguration** » plutôt que de reprogrammation. On modifie des connexions ou le comportement du composant. Il s'agit bien de **réseaux logiques reconfigurables et modifiables**
- Un circuit logique programmable est composé de nombreuses **cellules logiques** élémentaires et **bascules logiques** librement connectables
- Les cellules logiques sont connectées de manière **définitive** ou **réversible** par programmation, afin de réaliser les fonctions numériques voulues. L'intérêt est qu'**une même puce peut être utilisée dans de nombreux systèmes électroniques différents**

- Performances



Exemple:

- Processeur à 1 GHz
  - 1 000 000 000 opérations par seconde
- FPGA à 300 MHz avec 1000 multiplicateurs
  - 300 000 000 000 opérations par seconde

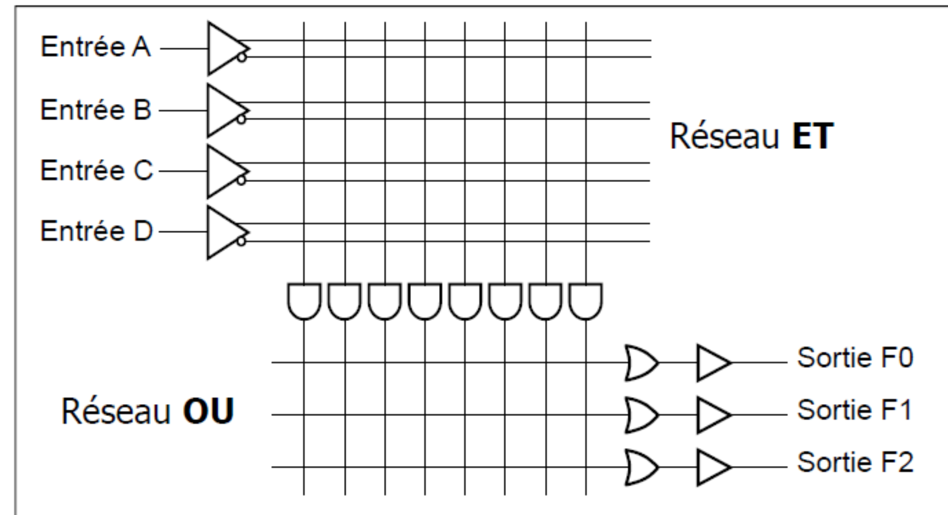
Classification des Circuits Logiques Programmables

Une fonction logique peut-être exprimée par une équation logique :

- **F = somme de produits**
- Exemple : F0 = (D and B and A) or (notC and A) or (D and notC)

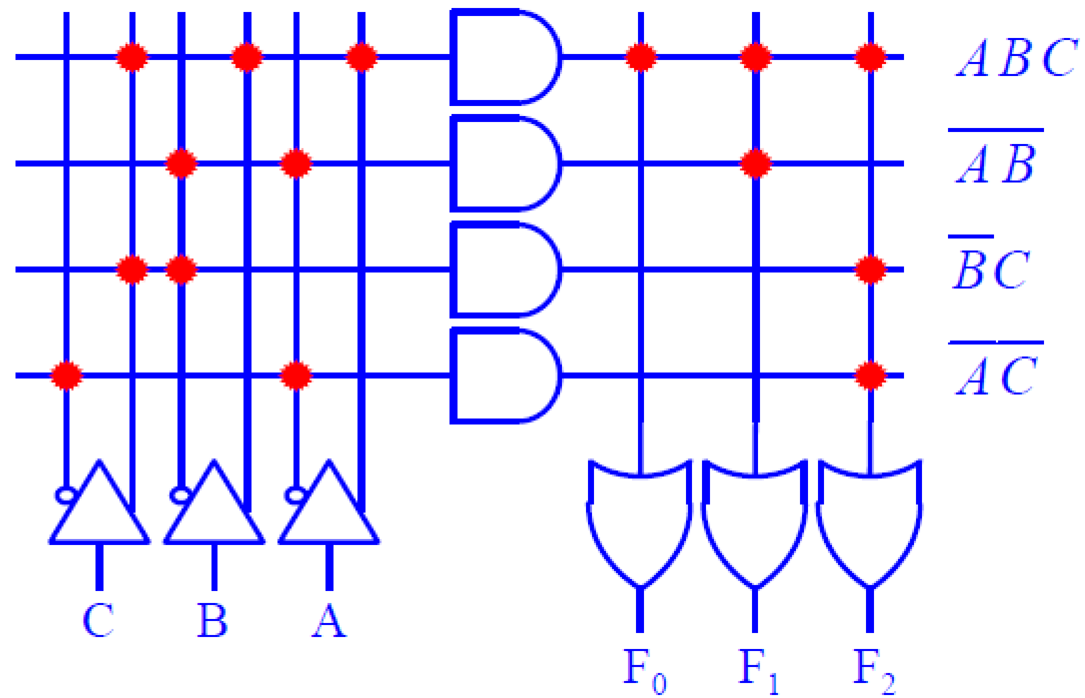
Circuit universel :

- ***<u>Un réseau de ET suivi d'un réseau de OU</u>***



Famille	Réseau ET	Réseau OU	Circuits
FPLA Field Prog. Logic Array	Programmable	Programmable	GAL
PAL Prog. Array Logic	Programmable	Fixe	SPLD, CPLD
FPGA Field Prog. Gate Array	Programmable	Inexistant	FPGA
PLE Prog. Logic Element	Fixe	Programmable	PROM

Structure des Circuits Programmables: Exemple



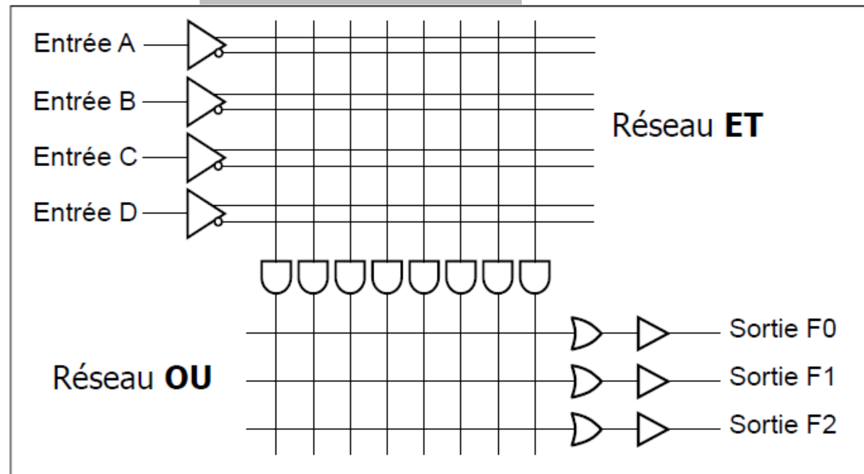
$$F_0 = ABC$$

$$F_1 = ABC + \overline{AB}$$

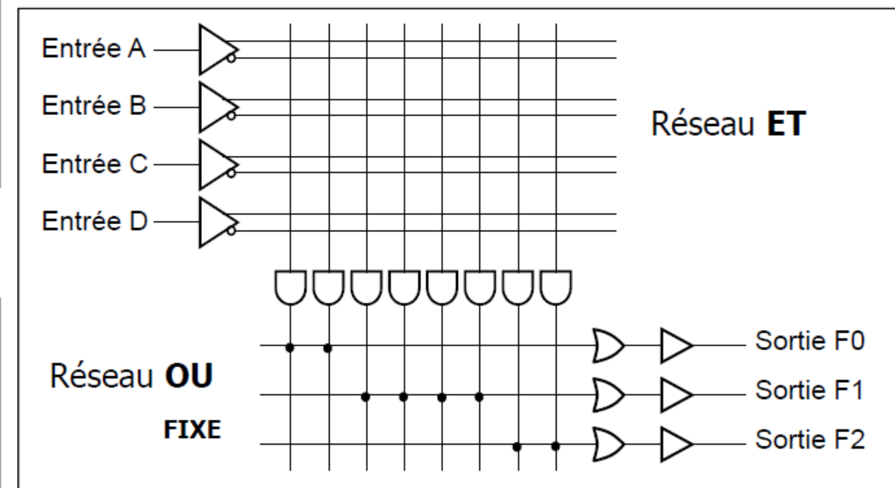
$$F_2 = ABC + \overline{B}C + \overline{A}\,\overline{C}$$

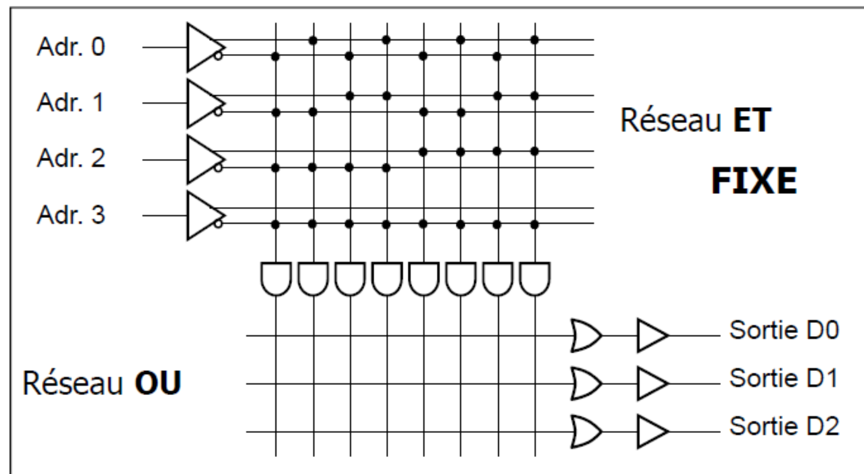
Familles des Circuits Universels (PLD)

FPLA (GAL)



PAL (SPLD)

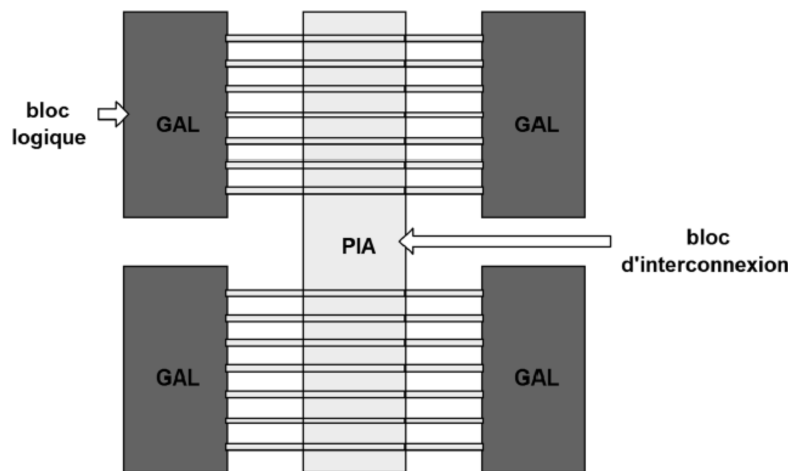


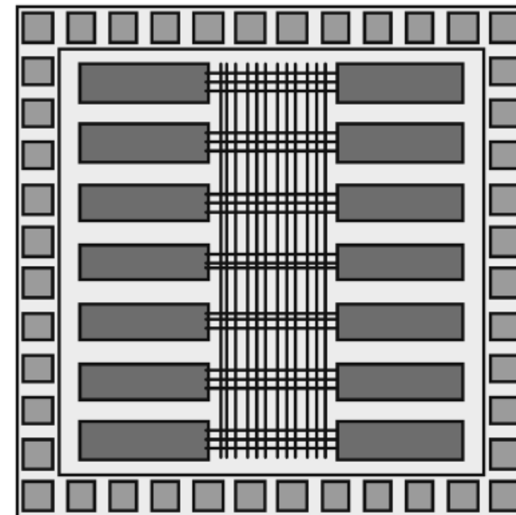


PLE (PROM)

CIRCUIT CPLD: ARCHITECTURE

- Les **CPLD** (comme le GAL, PAL, EPLD), de conception **plus ancienne**, utilisent des « **macrocellules** » logiques, composées d'un réseau combinatoire de portes ET et OU afin d'implémenter des équations logiques. Des bascules sont disponibles seulement dans les blocs d'entrée-sortie. Un composant contient de **quelques dizaines à quelques centaines** de macrocellules
- Comme le routage est fixe, l'utilisation **des ressources n'est pas optimale** (tout terme non utilisé dans une équation logique équivaut à des portes perdues), avec des taux d'utilisation d'environ **25 %**
- On distingue les CPLD des autres PLD car ils contiennent l'équivalent de plusieurs composants PLD, reliés par **une matrice d'interconnexion**

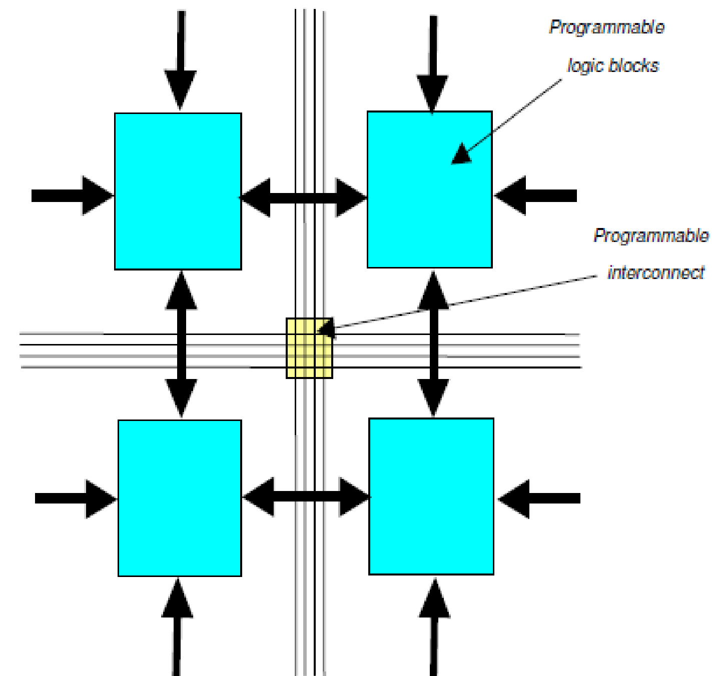




Les FPGA

LES FPGA

- Les FPGA sont initialement destinés au **prototypage** de systèmes numériques complexes. Ils sont une **bonne alternative** aux circuits spécifiques, les ASIC (Application Specific Integrated Circuit), pour des petites ou moyennes séries
- Il existe plusieurs grands fabricants : ALTERA et XILINX, …
- Chaque fabricant propose des composants de taille variable : de 100.000 à 10.000.000 portes logiques
- Quelque soit la technologie utilisée, aucune porte logique n'est réellement implantée. Il s'agit en fait de **blocs logiques programmables** et d'une mer de **connexions programmables**. Chez Xilinx, ces blocs logiques sont appelés CLB (Common Logic Blocks)



LES FPGA

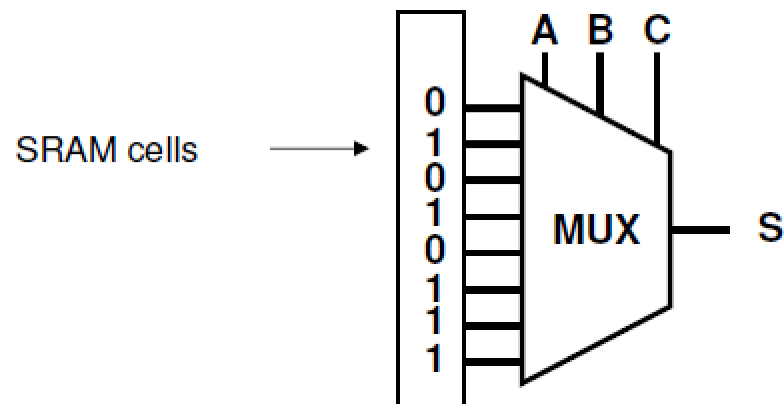
- La plupart des grands FPGA sont fondés sur **des cellules SRAM** aussi bien pour le **routage** du circuit que pour les **blocs logiques** à interconnecter
- Un **bloc logique** est de manière générale constitué d'une table de correspondance (**LUT ou Look-Up-Table**) et d'une **bascule** (Flip-Flop). La LUT sert à implémenter des équations logiques ayant généralement 4 à 6 entrées et une sortie. La bascule permet de mémoriser un état (machine séquentielle) ou de synchroniser un signal (pipeline)
- Les blocs logiques, présents en grand nombre sur la puce (de quelques milliers à quelques millions), sont connectés entre eux par **une matrice de routage configurable**. Ceci permet la reconfiguration à volonté du composant, mais occupe une place importante sur le silicium et justifie le coût élevé des composants FPGA
- Les densités actuelles **ne permettent plus un routage manuel**, c'est donc un outil de placement-routage automatique qui fait correspondre le schéma logique voulu par le concepteur et les ressources matérielles de la puce

PRINCIPE DE LOOKUP TABLE (LUT)

- Exemple d'un LUT

Multiplexers

$S = A.B + C$

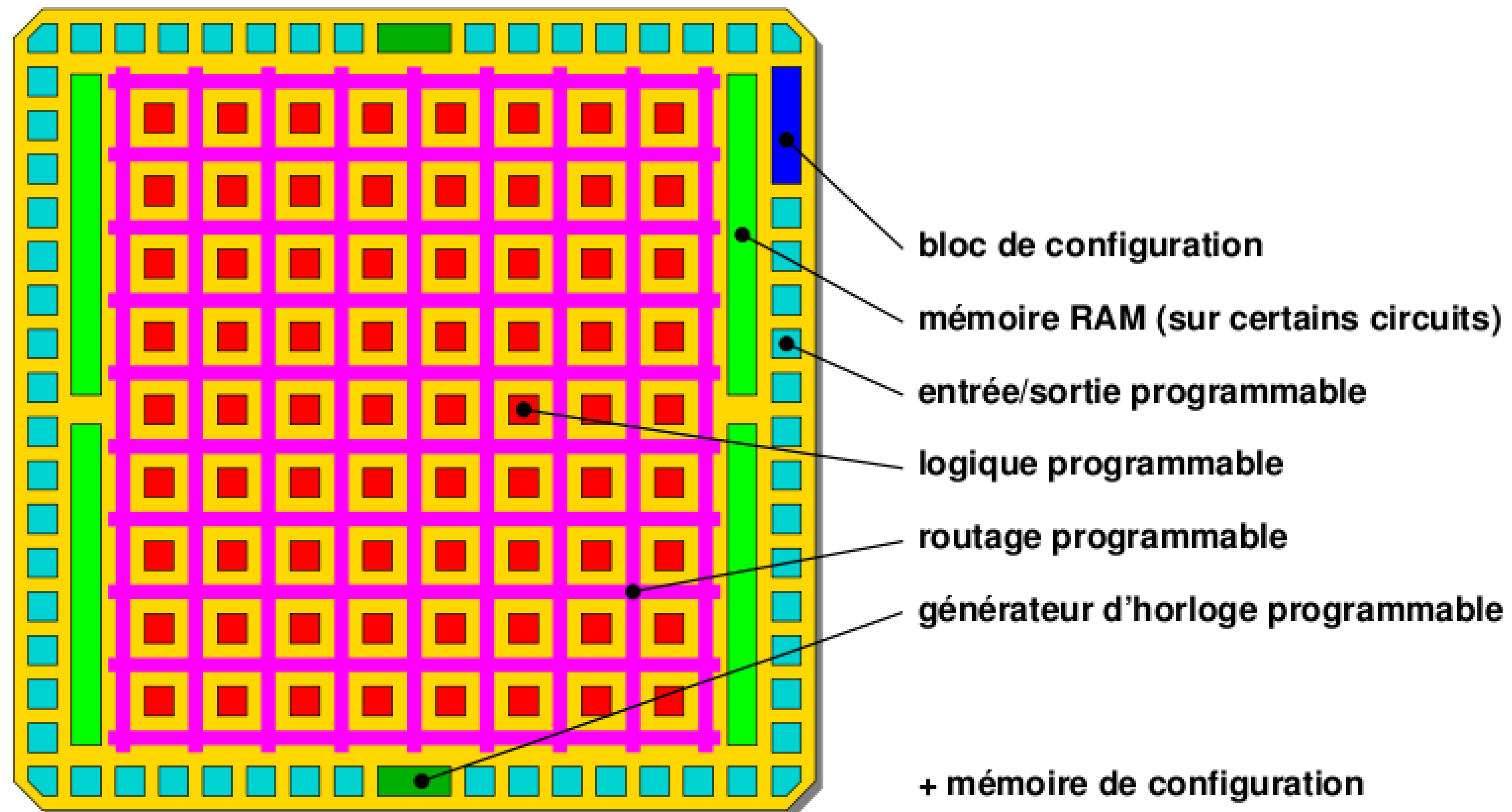


A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

LES FPGA

- Comme la configuration (**LUTs et routage**) est faite par des points mémoire volatiles, il est nécessaire de sauvegarder le design du FPGA dans **une mémoire non volatile externe**, généralement une mémoire Flash. Certains fabricants se distinguent toutefois par l'utilisation de cellules **EEPROM** pour la configuration, éliminant le recours à une mémoire externe
- Quelques fonctionnalités particulières disponibles sur certains FPGA :
  - blocs de mémoire supplémentaires (hors des LUT)
  - multiplieurs câblés (coûteux à implémenter en LUT) pour traitements DSP
  - cœur de microprocesseur embarqué comme par exemple des architectures PowerPC ou ARM
  - blocs PLL pour synthétiser ou resynchroniser les horloges
  - impédance contrôlée numériquement dans les entrées-sorties, évitant de nombreux composants passifs sur la carte
  - couche MAC Ethernet
  - …

EXEMPLE: STRUCTURE INTERNE (FPGA – XILINX)



Développement sur Les FPGA

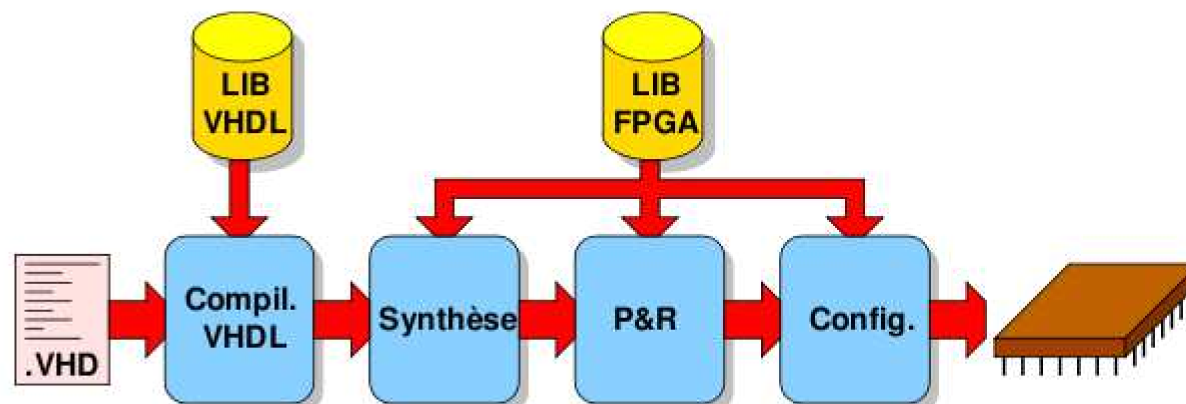
Langages de Description + Structure d'un Module VHDL

- Afin de pouvoir finaliser un FPGA, il est nécessaire d'utiliser un langage de description matériel ou bien un outil de saisie graphique. Après compilation de cette description, on obtient un fichier de configuration pour le FPGA choisi. VHDL et Verilog sont les deux langages de description les plus répandus

La description d'un système numérique par le biais du langage VHDL passe par 3 étapes différentes :

- la déclaration des ressources externes (bibliothèques)
- la description de l'entité du système, correspondant à la liste des entrées/sorties
- la description de l'architecture du système, correspondant à la définition des fonctionnalités du système

L'ensemble est contenu dans un fichier source portant l'extension *.vhd.



Structure d'un Module VHDL

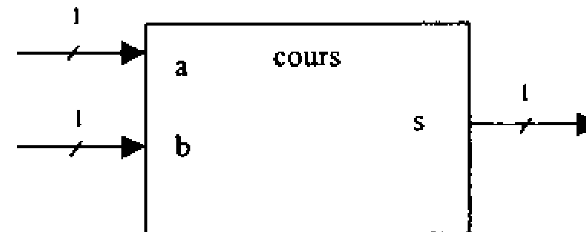
- **Déclaration des ressources externes**
  - Permet d'inclure des librairies de types prédéfinis ou fonctions
  - Réalisée automatiquement pour les bibliothèques courantes

On retrouve en en-tête du fichier source *.vhd les instructions suivantes :

```
l i b r a r y IEEE ;
use IEEE . STD LOGIC 1164 .ALL ;
use IEEE . STD LOGIC ARITH .ALL ;
use IEEE . STD LOGIC UNSIGNED.ALL ;
```

- **Entité**: Description des ports d'entrées / sorties (avec sa direction - in, out, inout - et son type)

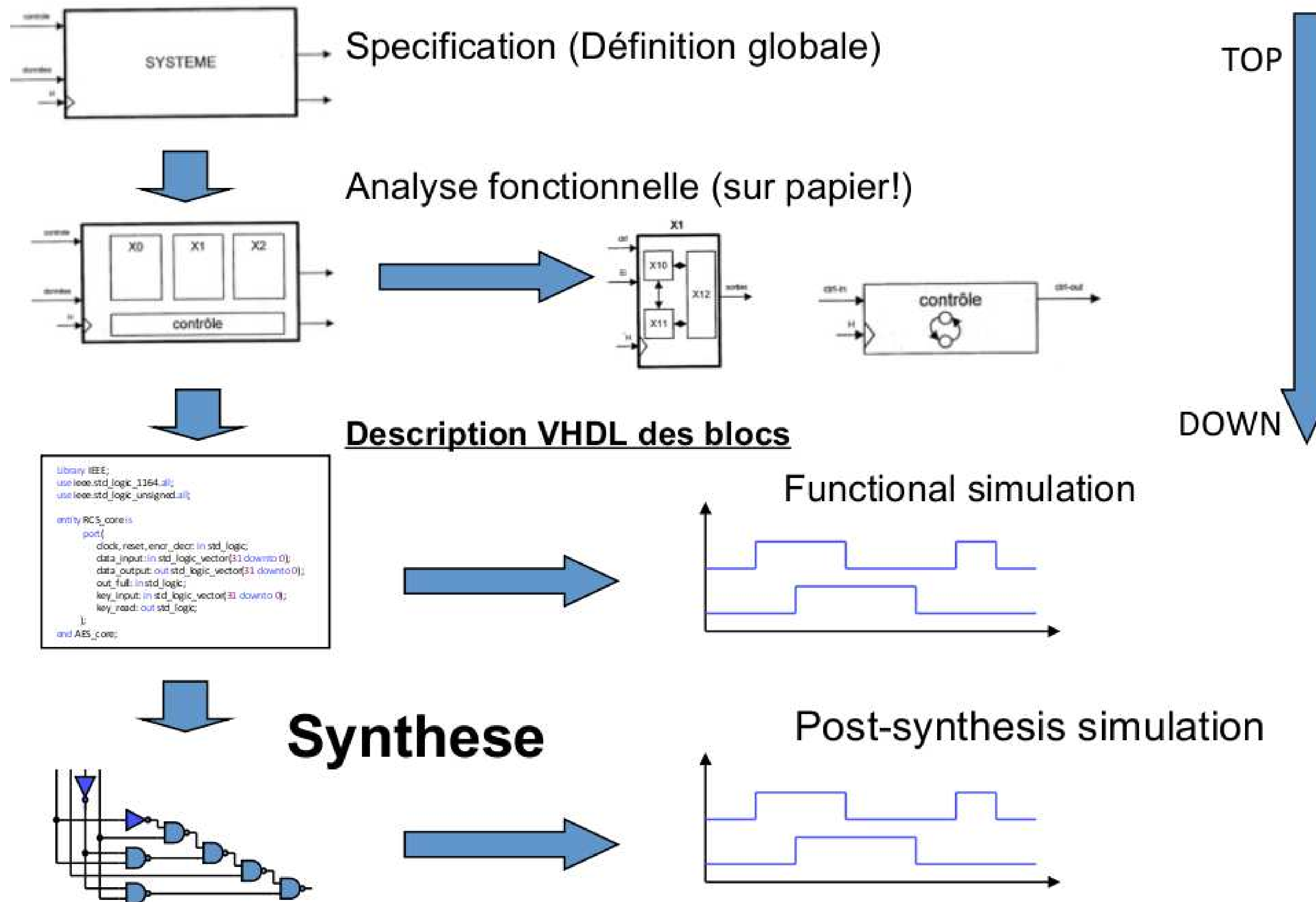
```
entity cours is
port (
a , b : in STD LOGIC ;
s : out STD LOGIC
) ;
end cours ;
```



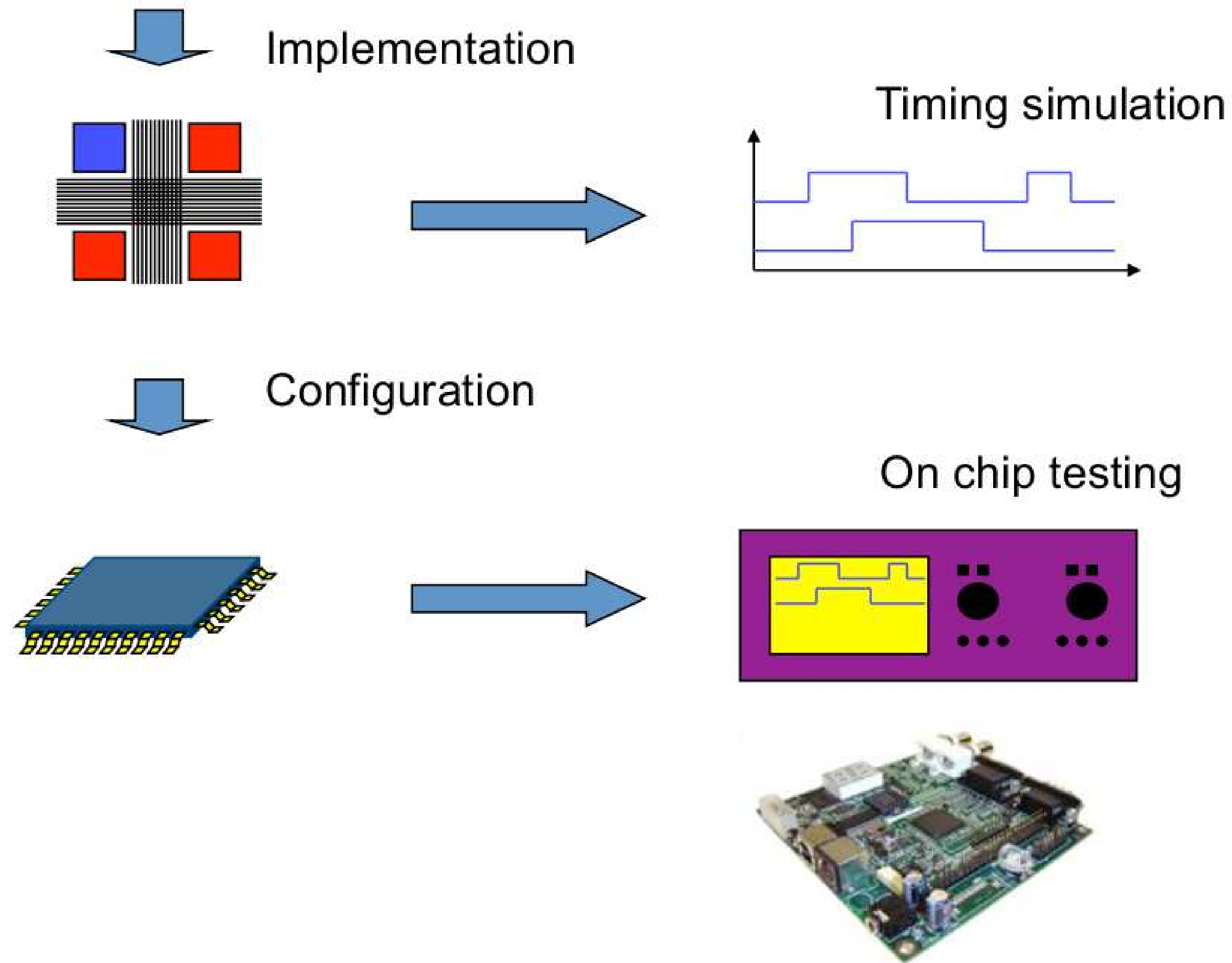
- **Architecture**: Reliée à une entité. Description du fonctionnement du système : Comportementale + Structurelle

```
architecture Behavorial of cours is
-- declaration des signaux
begin
processus1 ;
processus2 ;
. . .
end Behavorial ;
```

PHASES DE DÉVELOPPEMENT (1)



Phases de Développement (2)



LES ASIC

Application Specific Integrated Circuit

- Un ASIC (*Application-Specific Integrated Circuit*, ,circuit intégré développé pour un client) est un circuit intégré spécialisé. En général, il regroupe un grand nombre de fonctionnalités uniques ou sur mesure
- L'intérêt de l'intégration est de réduire les coûts de production et d'augmenter la fiabilité.
  - Avantages:
    - un contrôle total du produit
    - un coût de production réduit
  - Inconvénients :
    - un coût de développement élevé (notamment pour la fabrication des masques de gravure)
    - un délai de développement de plusieurs mois

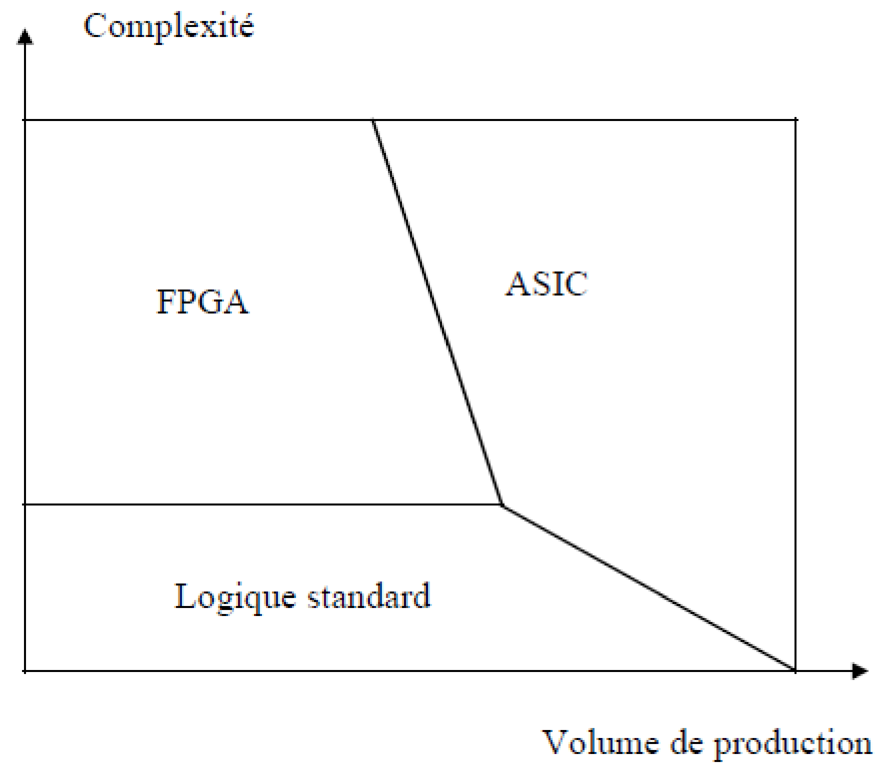
Application Specific Integrated Circuit

- Le développement d’un circuit électronique numérique se fait en utilisant un langage de description (VHDL, Verilog ou encore SystemC), qui est ensuite compilé par synthèse logique pour produire automatiquement le dessin du circuit. Les mêmes langages de description sont utilisés pour réaliser des prototypes avec des composants logiques programmables (FPGA par exemple)
- En raison du coût initial important, la production d'ASIC est généralement réservée à de gros volumes (>100 000 pièces par an). Dans le cas contraire, le fabricant rattrape le coût de son investissement par un prix de vente plus élevé
- On qualifie les gros ASIC de SoC (*System-on-Chip*, ou système sur silicium), lorsqu'ils intègrent processeur(s), interfaces, mémoires, etc., totalisant plusieurs millions de portes logiques, et qu'ils assurent la quasi-totalité des fonctions de la carte

CONCLUSION: PLD VERSUS ASIC

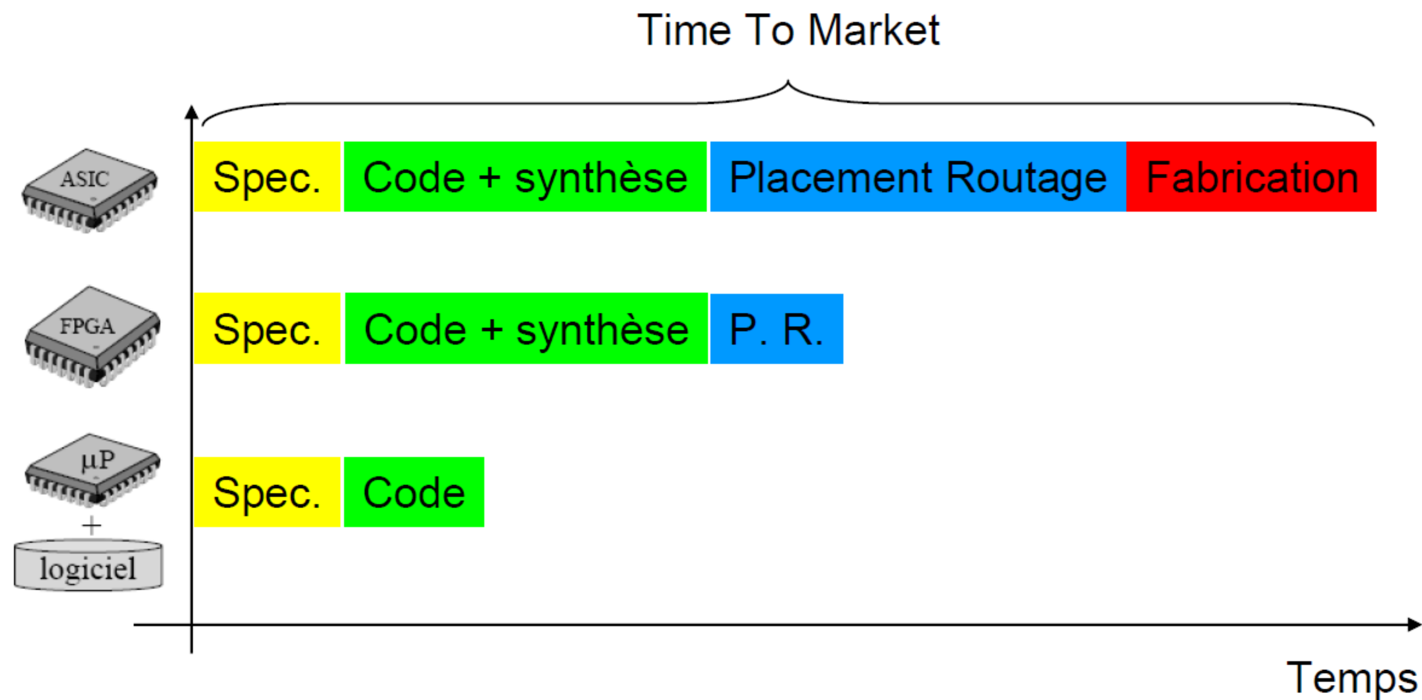
PERFORMANCES: FPGA VS ASIC

- Complexité (nombre de portes)/volume de production



Importance du «Time To Market »

- Le cahier des charges du développement d'un système doit prendre en compte les paramètres suivants :
  - Le coût de production
  - Les performances
  - La consommation
  - L'intégration
  - L'évolution du produit



DIFFÉRENCES ENTRE UN ASIC ET UN PLD

ASIC
Application Specific Integrated Circuits

- Choix du fondeur
- Conception du circuit (full-custom - bibliothèques)
- Très grand niveau d'intégration
- Fabrication à très grand nombre d'exemplaire

• Un circuit dédié à une application
• Choix de la technologie
⇨ Maximum de performances

PLD
Programmable Logic Device

- Choix du circuit
- Programmation du circuit (logiciel + interface circuit)
- Intégration limitée
- Implémentation nombre d'exemplaires limité

• Un circuit programmé pour une application
• Technologie figée mais :
⇨ Facilité de programmation